Towards an Autonomic Computing Environment

Towards an Autonomic Computing Environment

Roy Sterritt[1] Dave Bustard[2]
[1]*School of Computing and Mathematics*
[2] *School of Computing and Information Engineering*
Faculty of Informatics
University of Ulster
Northern Ireland
{r.sterritt, dw.bustard}@ulster.ac.uk

Abstract

Autonomic Computing is a promising new concept in system development. It aims to (i) increase reliability by designing systems to be self-protecting and self-healing; and (ii) increase autonomy and performance by enabling systems to adapt to changing circumstances, using self-configuring and self-optimizing mechanisms. This paper discusses the type of system architecture needed to support such objectives.

1. Introduction

Computing systems are expected to be *effective*. This means that they serve a useful purpose when they are first introduced and continue to be useful as conditions change. *Autonomic Computing*, launched by IBM in 2001 [1], is emerging as a valuable new approach to the design of effective computing systems.

The autonomic concept is inspired by the human body's autonomic nervous system. By analogy, humans have good mechanisms for adapting to changing environments and repairing minor physical damage. It is hoped that computing systems can be developed with similar properties.

It is likely that many branches of computer science research and development will contribute to progress in autonomic computing. In particular, it brings together work in software engineering and artificial intelligence [2]. Research on dependable systems should be especially influential, as dependability covers many relevant system properties such as *reliability*, *availability*, *safety*, *security*, *survivability* and *maintainability* [3]-[5].

This paper discusses the general architecture of an autonomic system. It first clarifies the basic requirements and activities of such systems and then considers possible supporting elements. These ideas combine existing suggestions for autonomic system structure with work in other research areas.

2 Autonomic System Architecture

Figure 1 summarizes the general properties of autonomic systems [5]. Essentially, the objectives represent broad system requirements while the attributes identify basic implementation mechanisms.

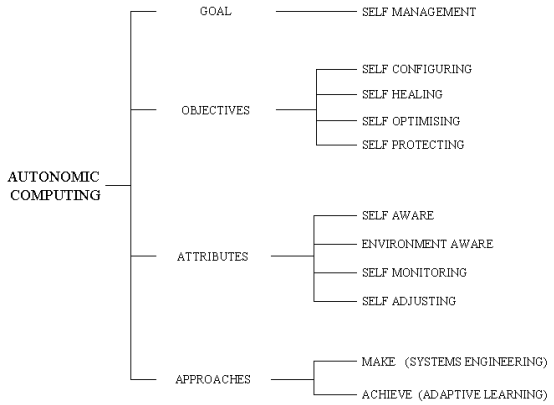

Figure 1 Autonomic Computing Tree

An autonomic system is *self-managing,* meaning that it is *self-protecting*, *self-configuring*, *self-healing* and *self-optimizing*.

Self-healing is concerned with ensuring effective recovery when a fault occurs. This means successfully identifying the fault and then, where possible, repairing it. Also, there should be minimal disruption to users, avoiding loss of data and significant delays in processing.

Self-optimization means that a system is aware of its ideal performance, can measure its current performance against that ideal and has strategies for attempting improvements.

Proceedings of the 14th International Workshop on Database and Expert Systems Applications (DEXA'03)
1529-4188/03 $17.00 © 2003 IEEE

A self-protecting system will defend itself from accidental or malicious external attack. This means being aware of potential threats and having ways of handling those threats. This may include self-healing actions if an attack is successful, and perhaps some self-optimization to increase protection.

Finally, self-configuring is a system's ability to readjust itself automatically to changing circumstances. This may simply be in support of ongoing development or to assist in self-healing, self-optimization or self-protection.

To achieve these objectives a system must be aware of its internal state (*self-aware*) and current external operating conditions (*environment-aware*). Changing circumstances are detected through *self-monitoring* and adaptations are made accordingly (*self-adjusting*). In more detail, this means a system having knowledge of its available resources, its components, their desired performance characteristics, their current status, and the status of inter-connections with other systems.

The ability to operate in a heterogeneous environment requires the use of open standards to understand and communicate with other systems.

Further information on autonomic computing can be found in IBM's autonomic 'manifesto' [1] and subsequent 'blueprint' [6].

2.1 Autonomic Artifacts

It is assumed that an autonomic computing system is made up of a connected set of *autonomic elements*. Each element must include *sensors* and *effectors* [7]. Monitoring behavior through the sensors, comparing this with expectations, deciding what action, if any, is needed and then executing that action through effectors, creates a control loop (Figure 2)[8].

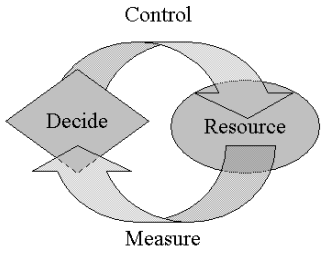

Figure 2 Control Loop [8]

Figure 3 shows a possible system architecture to support this model.

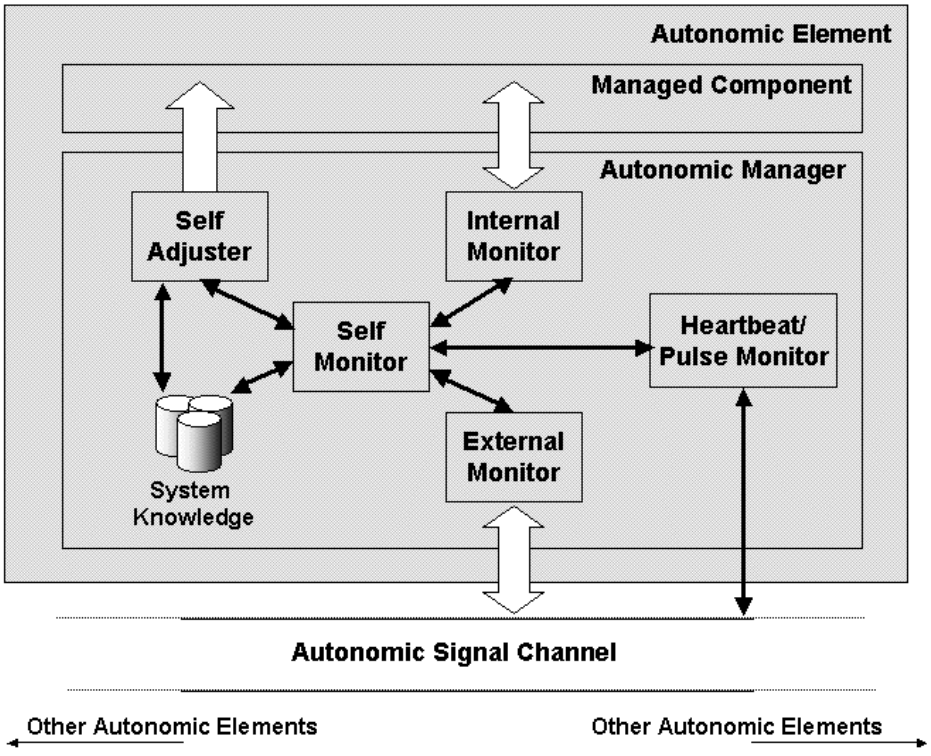

Figure 3 Potential Architecture of an Autonomic Element

The IBM autonomic 'blueprint' [6] assumes that in each autonomic element there is *managed component* and a corresponding *autonomic manager* implementing the required self-monitoring and self-adjusting.

An *internal monitor* observes the state of the managed component and passes this information to the *self monitor* for evaluation and action. The measured state is compared with the expected state held in a *system knowledge base*. Undesirable deviations are reported to the *self adjuster* for action, which may result in changes to the *managed component*. Similarly, an *external monitor* observes the state of the environment via an *autonomic signal channel* and this also may trigger internal changes. The signal channel provides linkage to other autonomic managers. These may be virtual (in the same physical system), peer-to-peer or networked [9].

The *heartbeat or pulse monitor* provides a summary of the state of an autonomic element to other autonomic elements responsible for monitoring that state.

Figure 4 suggests how autonomic elements might be connected. The artifacts within an autonomic element (Figure 3) and autonomic elements within a system (Figure 4) collaborate using asynchronous communicating techniques, like a message bus [6]. Care must be taken in designing the monitoring protocol to ensure that the monitoring activity and traffic are maintained at acceptable levels.

There are a number of ways in which the 'health' of autonomic elements might be monitored. One is to have dedicated elements for that purpose (network managers). Another is distribute monitoring responsibility among the autonomic elements so that they monitor each other. This will increase robustness but is more difficult to manage.

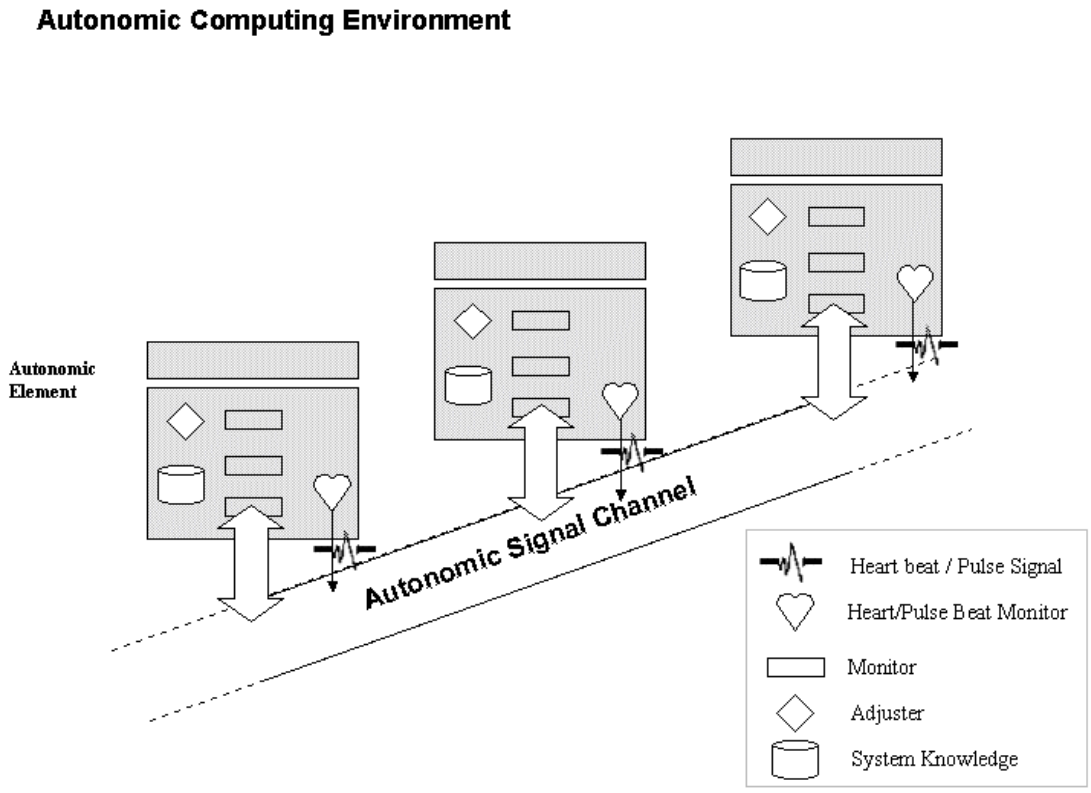

Figure 4 Autonomic Computing Environment

In some respects, achieving autonomic computing on servers will be an easier task than on clients. Servers are likely to have received the level of investment to ensure in-built fault tolerance and include extensive redundancy – including facilities such as 'hot swapping'. Clients are often machines built on the *faster, cheaper and smaller* philosophy with limited, if any redundancy. Servers are also likely to have a user base of highly skilled teams, whereas clients are often in the personal computing domain. Other considerations are required for personal computing such as flexibility of location (e.g. laptops) and of hardware (e.g. palm devices) and software configuration that complicate further the goal of achieving autonomic computing [9].

2.2 Reflexes and Healing

A concept inspired by biological systems is the duel approach of reflexes and healing [10]. Animals have a reflex system, where the nerve pathways enable rapid response to pain. Reflexes cause a rapid, involuntary motion, such as when a hot surface is touched. The effect is that the system reconfigures itself, moving away from the danger to keep the component functioning.

On a much longer timescale, the body will heal itself. Resources from one part of the system are redirected to rebuild the injured body part, including repair of the reflex response network. While this cannot help in the real-time response, directly after an event, it can prepare the system for the next event. In addition, it can readjust the system for operation with a reduced set of resources [10].

2.3 Heart Beat Monitor

The heartbeat monitor is a specific type of environment-awareness (Figure 5). A similar feature has been identified in the computational grid domain. The OGSA (open grid services architecture) has a facility referred to as the *Globus Heartbeat Monitor* which is designed to detect and report processes that fail to provide a 'heartbeat' [11].

The heartbeat monitor function ensures an element is operating at a basic level. More information is required, however, to determine how well an element is performing, if it is necessary to improve its operation, consistent with the needs of autonomic computing.

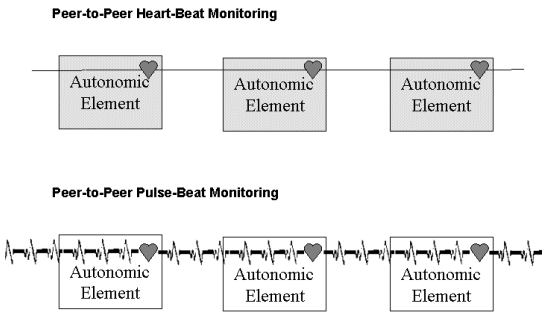

Figure 5 Heartbeat or Pulse-beat Monitoring

2.4 NASA's Beacon Monitor

NASA has shown a growing interest in adaptive operations and onboard autonomy [12]. NASA missions, particularly those to deep space, are considering autonomic decision making to avoid the unacceptable lag time between a craft encountering new situations and the round-trip delay in obtaining guidance from mission control. Two of the first notable missions to use autonomy are DS1 (Deep Space 1) and the Mars Pathfinder [13].

One of the interesting outcomes from the DS1 mission work was the *beacon monitor* concept [14]. With beacon monitoring, the spacecraft sends a signal to the ground that indicates how urgent it is to track the spacecraft for telemetry. This concept involved a paradigm shift for NASA from routine telemetry downlink and ground analysis to onboard health determination and autonomous data summarization [14].

In high-level concept terms, the beacon monitor is similar to the heartbeat monitor in grid computing, with the addition of a tone to indicate the degree of urgency involved. The following table summarises the tone definitions [15]:

Nominal	All functions as expected no need to downlink.
Interesting	Interesting – non-urgent event. Establish comms when convenient.
Important	Comms need to take place within timeframe or else state could deteriorate.
Urgent	Emergency. A critical component has failed. Cannot recover autonomously and intervention is necessary immediately.
No Tone	Beacon mode is not operating.

A hybrid approach for the autonomic environment [16] is to use the urgency concept of the beacon monitor to turn the heartbeat monitor into a *pulse beat monitor*– so instead of just checking the presence of a beat, the rate is also measured (Figure 5).

This effectively provides a reflex reaction within the environment, sharing responsibility for environment monitoring and indicating increasing urgency levels.

3. Conclusion

Autonomic computing is an emerging holistic approach to computer system development that aims to bring a new level of automation and dependability to systems through self-healing, self-optimizing, self-configuring and self-protection functions.

The promotion of autonomic computing will be assisted by good examples of its use. An important step is that direction is the establishment of design patterns for such systems.

This paper has presented a general design template based on a simple characterization of autonomic systems, incorporating ideas from related research areas. This involves internal and external monitoring, and consequential adjustment to improve system operation. The notion of a pulse monitor is used to provide a simple means of observing the 'health' of each autonomic element. A demonstration system to illustrate and further refine this architecture us currently under development.

Acknowledgements

This work was undertaken through the Centre for Software Process Technologies, which is supported by the EU Programme for Peace and Reconciliation in Northern Ireland and the Border Region of Ireland (PEACE II).

References

[1] P. Horn, "Autonomic computing: IBM perspective on the state of information technology", IBM T.J. Watson Labs, NY, 15th October 2001. Presented at AGENDA 2001, Scotsdale, AR. (available http://www.research.ibm.com/autonomic/), 2001

[2] IJCAI Workshop, "AI and Autonomic Computing: Developing a Research Agenda for Self-Managing Computer Systems, Acapulco, Mexico, August 10, 2003, http://www.research.ibm.com/ACworkshop

[3] A. Avizienis, J.-C. Laprie, B. Randell, "Fundamental Concepts of Dependability", UCLA CSD Report #010028, 2000.

[4] B. Randell, "Turing Memorial Lecture – Facing Up to Faults", Comp. J. 43(2), pp 95-106, 2000.

[5] R Sterritt, DW Bustard, "Autonomic Computing-a Means of Achieving Dependability?", Proceedings of IEEE International Conference on the Engineering of Computer Based Systems (ECBS'03), Huntsville, Alabama, USA, April 7-11 2003, pp 247-251.

[6] IBM, "An architectural blueprint for autonomic computing", April 2003.

[7] A.G. Ganek and T. A. Corbi, The dawning of the autonomic computing era, IBM Systems Journal, Vol 42, No 1, 2003, pp. 5-18

[8] Autonomic Computing Concepts, IBM White Paper, 2001.

[9] D.F. Bantz, C. Bisdikian, D. Challener, J.P. Karidis, S. Mastrianni, A. Mohindra, D.G. Shea, M. Vanover, Autonomic personal computing, IBM Systems Journal, Vol 42, No 1, 2003, pp. 165-176

[10] T. Bapty, S. Neema, S. Nordstorm, S. Shetty, D. Vashishtha, J. Overdorf, P. Sheldon, "Modeling and Generation Tools for Large-Scale, Real-Time Embedded Systems", 10th IEEE International Conference and Workshop on the Engineering of Computer-Based Systems, Huntsville, Alabama, USA, 7-10th April 2003, pp11-16.

[11] The Globus Heartbeat Monitor Specification v1.0, http://www-fp.globus.org/hbm/heartbeat_spec.html

[12] J. Wyatt, R. Sherwood, M. Sue, J. Szijjarto, "Flight Validation of On-Demand Operations: The Deep Space One Beacon Monitor Operations Experiment", 5th International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS '99), ESTEC, Noordwijk, The Netherlands, 1-3 June 1999

[13] N. Muscettola, P. P. Nayak, B. Pell, and B. Williams, "Remote Agent: To Boldly Go Where No AI System Has Gone Before".. Artificial Intelligence 103(1-2):5-48, 1998.

[14] J.Wyatt, H. Hotz, R. Sherwood, J. Szijjaro, M. Sue, "Beacon Monitor Operations on the Deep Space One Mission", 5th Int. Sym. AI, Robotics and Automation in Space, Tokyo, Japan, 1998

[15] R. Sherwood, J. Wyatt, H. Hotz, A. Schlutsmeyer, M. Sue, "Lessons Learned During Implementation and Early Operations of the DS1 Beacon Monitor Experiment," Third International Symposium on Reducing the Cost of Ground Systems and Spacecraft Operations, Tainan, Taiwan, 1999.

[16] R. Sterritt, "Towards Autonomic Computing: Effective Event Management", Proceedings of the 27th Annual IEEE/NASA Software Engineering Workshop, Greenbelt, MD, Dec. 2002,. pp 40-47.